Part IV
Client Security

Chapter 13 Securing Outlook

Technology—like science—is neutral all through, providing only means of control applicable to any purpose, indifferent to all.

—John von Neumann

E-mail-transmitted malware is an unfortunate fact of life. Malicious code attached to e-mail messages can contain worms or viruses; once one machine becomes infected, highly interconnected e-mail systems allow malware to spread very quickly. How'd we get here? Customers asked for advanced customization features, so Microsoft delivered them as part of the Microsoft Office system. The first really bothersome e-mail viruses were created and spread using the Office macro language; more recently, the mix of malware has shifted toward attacks that exploit vulnerabilities in Microsoft Internet Explorer. The integration between Microsoft Outlook, other Office applications, and the Microsoft Windows operating system delivers a great set of benefits, but some malicious people have used those features for ill—no different than almost any other technology. Completely disallowing all forms of scripts and executables is a cure worse than the disease; the ideal remedy would be to teach users not to open untrusted attachments. However, that assumes that users will always do what you tell them to do, and we know better. Fortunately, a combination of Outlook features and administrative savvy can be applied to minimize, if not eliminate, the vulnerability.

Of course, e-mail-borne malware gets the lion's share of press attention, but Outlook offers many useful security features, notably support for the Secure/Multipurpose Internet Mail Extensions (S/MIME) message security protocol, plus a variety of smaller security tweaks that you can apply to tighten your desktop users' security.

Understanding Outlook's Security Features

There's often a tension between convenience and security, and that's particularly true of the security features introduced in the Outlook E-Mail Security Update for Microsoft Outlook 98 and Outlook 2000. (The update's features are built in to Microsoft Outlook 2002 and Outlook 2003.) The goal of the update was to add features to Outlook to limit the spread of e-mail-borne malware; among other things, this required restricting users' ability to access some kinds of attachments, like executable files and VBScripts. In addition, the security update causes Outlook to warn users when external programs (from both Microsoft and third parties) try to access certain properties and methods; this has had a much greater impact than the attachment security changes in some corporate environments. Fortunately, administrators can customize the update's behavior through a Microsoft Exchange Server 2003 public folder, and end users have some ability to customize attachment handling on systems where administrators aren't imposing control.

The Outlook Security Update

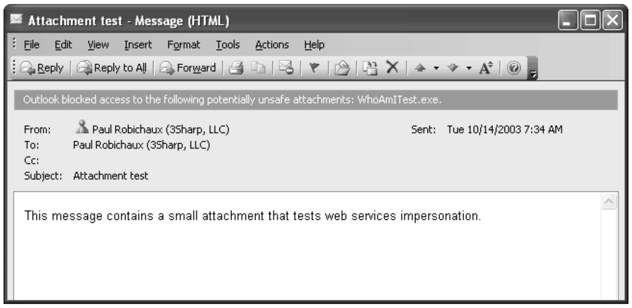
The security update (which is what I'm going to call it, even though it's included in the current version of Outlook) includes five major changes:

- Improved attachment security. Outlook blocks access to some file types altogether, including .exe and .pif files and screen savers. Administrators can specify a second, less restricted set of file types that can't be opened directly, but can be saved to disk.
- The ability for users to control programmatic access to the address book and to Outlook's mail-sending functionality.
- Support for letting Exchange administrators specify which sources for code and Component Object Model (COM) add-ins for Outlook should be trusted. Note that this feature is only available in Outlook 2002 and Outlook 2003; it's not present in the security updates for Outlook 2000 and Outlook 98. These restrictions apply only to COM add-ins, not to programs that use Messaging Application Programming Interface (MAPI) or Collaboration Data Objects (CDO).
- A change to the default security zone in which Outlook runs.
- Code on unpublished or one-off Outlook forms does not run unless specifically allowed by the Exchange administrator.

Attachment Security

As a technical solution to what is largely a behavioral problem, Outlook 2002 checks the file type of each message attachment against an internal list of file types. A default list is included with the product, as shown here, but you can override or customize this list using an Exchange public folder.

- **Level 1** These file types (including .bat, .exe, .vbs, .lnk, and .js) are blocked by Outlook. Recipients get a warning InfoBar listing the blocked files (see Figure 13-1) when they open or preview a message with a Level 1 attachment, but they can't see or access the attachment themselves (at least through Outlook; clients using Post Office Protocol [POP] and Internet Message Access Protocol [IMAP] clients other than Outlook can still get to Level 1 files). Table 13-1 lists the Level 1 attachment types.


Attachment test - Message (HTML)
File Edit View Insert Format Tools Actions Help
Reply Reply to All Forward

Outlook blocked access to the following potentially unsafe attachments: WhoAmITest.exe.

From: Paul Robichaux (3Sharp, LLC) Sent: Tue 10/14/2003 7:34 AM
To: Paul Robichaux (3Sharp, LLC)
Cc:
Subject: Attachment test

This message contains a small attachment that tests web services impersonation.

Figure 13-1. *Blocked attachments are still in the store, but users can't access them through Outlook.*

Table 13-1. Level 1 File Attachment Types

Extension	File Type
.ade	Microsoft Access project file
.adp	Access project
.app	Microsoft Visual FoxPro application
.bas	Microsoft Visual Basic class module
.bat	MS-DOS/Windows batch file
.cer	PKCS #7-encoded X.509 certificate file. Microsoft's documentation doesn't list this as a blocked file type, but it is.
.chm	Compiled Hypertext Markup Language (HTML) help file
.cmd	Windows NT, Windows 2000, or Windows XP batch file
.com	MS-DOS program
.cpl	Control Panel
.crt	PKCS#7-format digital certificate
.csh	C shell script file for Microsoft Services for UNIX
.exe	x86 executable
.fxp	Visual FoxPro compiled program
.hlp	Help file
.hta	HTML program
.inf	Setup information file
.ins	Internet Naming Service file
.isp	Internet communication settings file
.js	JavaScript script
.jse	Encoded JavaScript file
.ksh	Korn shell script file for Microsoft Services for UNIX
.lnk	Shortcut
.mda	Access add-in
.mdb	Access database
.mde	Encoded Access database
.mdt	Access workgroup information

(continued)

Table 13-1. Level 1 File Attachment Types *(continued)*

Extension	File Type
.mdw	Access workgroup information
.mdz	Access wizard
.msc	Microsoft Management Console (MMC) console file
.msi	Windows Installer package
.msp	Windows Installer patch
.mst	Visual Test source file
.ops	Office settings profile (generated by the Custom Installation Wizard)
.pcd	Visual Test compiled script
.pif	Shortcut to MS-DOS program
.prf	Outlook profile
.prg	FoxPro program
.pst	Outlook personal folders file
.reg	Registry keys for REGEDIT
.scf	Explorer command file
.scr	Screen saver
.sct	Windows scripting component
.shb	Shortcut to a document section
.shs	Shell scrap object
.url	Internet shortcut/Uniform Resource Locator (URL)
.vb	VBScript file
.vbe	Encoded VBScript file
.vbs	VBScript script
.wsc	Windows script component
.wsf	Windows script
.wsh	Windows Scripting Host settings file
.xsl	Extensible Markup Language (XML) style sheet

- **Level 2** There are no Level 2 file types by default; you have to add them yourself. With Level 2 attachments, you can see the icon for the attachment, and when you double-click it, you are prompted to save the attachment to your hard disk, but you can't run it directly from its current location. After you have saved the attachment, you can decide how to handle it. This is supposed to make users think before blindly double-clicking every collection of bits that lands in their Inbox.

When you attach a file to an outgoing message, Outlook checks the file type against the Level 1 list. If you've attached any Level 1 files, a dialog box warns you that the recipients might not be able to open the attachment. Clicking Yes in this dialog box sends the message as is. Note that you can tell Outlook to not give you this warning; I'll tell you how later.

When you receive a message that contains a Level 1 attachment, your Inbox displays the paper clip in the attachment column to let you know that the message includes an attachment. When you open an e-mail message containing an attachment, the attachment is blocked, and the Outlook InfoBar warns you that the attachment is untouchable. The File | Save Attachments command (as well as the View Attachments command on the shortcut menu that opens when you right-click) shows only those attachments that aren't blocked, rendering the others completely inaccessible. When you open the message itself, you'll see the same warning as shown in Figure 13-1, but you can still get to all attachments that have extensions that aren't on the banned list.

If you receive a message containing a Level 2 file as an attachment, the attachment appears normally. However, when you try to open it, you'll get a warning dialog box telling you that it's a bad idea to run the attachment directly and offering to let you save it to disk.

Address Book and Object Model Security

Outlook supports the Office object model, so you can write scripts and programs that automate repetitive actions. This is a double-edged sword: it's very useful to allow some programs (like synchronization tools for personal digital assistants [PDAs] or customer relationship management programs) to access contact information, but the same interfaces can be used by viruses or other malicious executables to propagate. In fact, many macro viruses invade the victim's address book to get addresses to which they can mail themselves; because the security update makes this harder, some virus creators have now switched to scanning local files and harvesting e-mail addresses from them.

To help counter this behavior, Outlook versions that include the Outlook Security Update 2003 turn on *object model guards* that restrict what outside applications can tell Outlook to do. There are three categories of object model guard: one category restricts calls made with the Simple Messaging Application Programming Interface (Simple MAPI; don't confuse Simple MAPI with Extended MAPI, which is not subject to the object model guard mechanism), one restricts calls made with the Outlook object model, and the third covers calls made using the Collaboration Data Objects (CDO) method. I describe the specific types of access you can guard against later in the chapter.

Security Zone Changes

In Outlook 2002 and Outlook 2003, the default security zone setting is Restricted Sites, rather than Internet. Within the Restricted Sites zone, active scripting is also disabled by default. This security zone disables most automatic scripting and prevents Microsoft ActiveX controls from opening without permission. This change is designed to protect against malware that might be contained in HTML messages. As long as you leave the default Outlook zone set to Restricted Sites, Outlook won't run scripts in HTML messages, and ActiveX controls in those messages are deactivated. You should ensure that you always apply Internet Explorer patches to all machines running Outlook, because Outlook uses Internet Explorer to render HTML messages.

S/MIME Security

S/MIME support is one of Outlook's unheralded important features. (I used to write S/MIME security software, so I admit to being a little biased.) Outlook's support for S/MIME gives you end-to-end protection: messages you create can be signed or encrypted on your computer, and they stay protected as they travel and while they are in the recipient's Exchange mailbox server. Encrypting, signing, decrypting, and verifying messages is easy, and Outlook gives you a great deal of control over the security settings used with particular certificates. One of the nicest things about Outlook's S/MIME support is that it's completely integrated (through CryptoAPI) with the rest of the system's cryptographic functions; if you're using certificates or smart cards for access control or other functions, you will probably be able to use the same certificates with Outlook (as long as the certificates they contain are marked by the issuer as usable with S/MIME).

Chapter 2, "Security Protocols and Algorithms," describes the S/MIME protocol in general terms, but this is a good time to get a bit more specific. The reason for the MIME in S/MIME is that the secured content in S/MIME messages is actually made up of Multipurpose Internet Mail Extension (MIME) body parts, as described in Request for Comments (RFC) 1847. That means, for instance, that a plaintext message can still contain an attached signature. This is called a *clear-signed message*, because the message is still readable by clients that don't understand S/MIME signatures. Its opposite, an *opaque-signed message*, contains the message and signature combined in a single part that cannot be read except by verifying the signature.

The S/MIME version 3 standard, which is what Outlook supports and uses by default, consists of several parts that define various aspects of how clients and servers should create, process, and manipulate secured mail, as follows:

- RFC 3369 describes the Cryptographic Message Syntax (CMS), the format of S/MIME messages. CMS is derived from the older Public Key Cryptography Standards (PKCS) #7 format (RFC 2315), which is why S/MIME-protected messages still appear to have attachments with the .p7m extension—that stands for PKCS #7 MIME part. This RFC is interesting primarily because it describes exactly how clients must sign, encrypt, decrypt, and verify messages and how the messages should be structured so that other clients can read them.
- RFC 3370 identifies the algorithms that all S/MIME version 3 software must support to be called S/MIME: Secure Hash Algorithm 1 (SHA-1) and Message Digest-5 (MD5) for hashing, Digital Signature Algorithm (DSA) and RSA for signatures, and RC2 and triple Data Encryption Standard (3DES) for message encryption. Individual implementations are free to add more algorithms to this set, provided they properly identify which algorithms a particular message uses so the recipient can figure it out.

- RFC 2632 describes certificate handling for S/MIME-compatible software. It specifies when and how clients should check for certificate revocation and expiration, how they should handle unknown certificate authority (CA)-specific extensions, and so forth.
- By its own admission, RFC 2633 "defines how to create a MIME body part that has been cryptographically enhanced according to CMS, which is derived from PKCS #7. This memo also defines the application/pkcs7-mime MIME type that can be used to transport those body parts."

As a messaging administrator, you might not ever want to read these RFCs, but they can give you some useful insight into why S/MIME clients act the way they do in some circumstances.

Outlook provides complete S/MIME version 3 support, plus some additional features defined in RFC 2634, including digitally signed message receipts, security labels (like secret or confidential), and a few other features of interest primarily to users of the Defense Messaging System (DMS), which I'm not going to cover in this book.

RPC over HTTPS

Exchange and Outlook use the remote procedure call (RPC) protocol to communicate. This is fine on local area networks (LANs), but most administrators wisely block RPC traffic at their network perimeter; there is no good reason to allow random Internet hosts to send you RPC packets—in fact, it's a good idea *not* to given the past history of vulnerabilities in the Windows RPC stack. This has posed a conundrum for Exchange administrators: what's the best way to allow remote users access to their mailboxes? There are several options to choose from: Microsoft Outlook Web Access does a good job overall, but doesn't allow access to stored mail while users are disconnected; POP and IMAP are useful lightweight protocols, but don't offer the full range of Exchange services; virtual private networks (VPNs) allow secure access, but they also allow the remote machine full run of the connected network, which isn't always desirable; and Internet and Security Acceleration (ISA) Server allows publishing RPC-based services while inspecting inbound RPC traffic to ensure its integrity and harmlessness.

In Outlook 2003, Microsoft has added full support for tunneling RPC packets inside of Hypertext Transfer Protocol (or, more precisely, Secure Sockets Layer [SSL]-protected HTTP) packets. With the right configuration (as described in Chapter 11, "Securing Internet Communications"), a mobile user can launch Outlook, connect to the corporate network on port 443, and have his or her RPC traffic tunneled from the network entry point to the Exchange server. Users get complete Outlook functionality, and administrators enjoy the protection of blocking plain RPC traffic at the perimeter. However, this magic requires some configuration on the Outlook side, which I discuss later in the chapter.

Information Rights Management

There's a popular saying that "information wants to be free." Although this is in many cases true, the *owner* of the information might not want it to apply to his or her confidential or proprietary information. Some organizations, like the U.S. government or Apple Computer, deal with this by imposing severe penalties on employees who leak sensitive materials. Most of us don't have that luxury, though, so it would be preferable to have some technological means to give information creators more control over where and how their documents and messages are used.

The Microsoft Office System works with Windows Rights Management Services (RMS) servers (which are built by installing the separate Windows Rights Management Services product on Microsoft Windows Server 2003) to provide a good set of information rights management (IRM) functionality. The goal behind IRM is simple: people who create documents or messages should be able to specify whether those documents can be modified, forwarded, or copied, and whether (and when) they should expire. The Microsoft rights management (RM) implementation uses the XML-based eXtensible rights Markup Language (XrML) to specify what rights the document creator has assigned; then it uses various cryptographic algorithms to securely embed those rights definitions in the document. RMS-protected documents can only be accessed with credentials from an RMS installation. Microsoft has also made available an Internet Explorer plug-in that allows viewing of protected documents from machines that aren't running the Office System.

A complete discussion of how IRM works is outside the purview of this book, because it involves setting up an RMS server and defining IRM usage policies for the organization. In this chapter, I limit my discussion to talking about how to make Outlook work with a functioning RMS server.

Customizing the Outlook Security Update

When Outlook starts up and logs on to an Exchange server, it looks for a registry key (shown later in Table 13-2) that tells it which version of a special public folder to look for. This folder can be named either Outlook Security Settings (which applies to Outlook 98 and Outlook 2000) or Outlook 10 Security Settings (which applies to Outlook 2002 and Outlook 2003). Based on what Outlook finds in the folder, it might use security settings that vary from the default. The contents of those public folders determine which settings Outlook uses; you post messages to the public folder using a special form.

If you want your users to have customized security settings, there's a registry key that must be set on each individual client, which you'll normally set as part of the deployment. When the key is present, Outlook looks on the server for custom security settings that apply to that user; if any exist, they are used. If the key is not present, the built-in security settings are applied to the computer. Microsoft chose to store these settings in a public folder because that provides a degree of security; as long as you set the folder permissions correctly, no one can sneak in new settings or change existing ones.

Installing the Security Package

The administrative tools for the Outlook Security Update consist of several files (including documentation) that are normally packaged as a single executable, Admpack.exe, which can be installed from the Office Resource Kit CD or the Office product CD. You can also download it from the Office Resource Kit Toolbox site at *http://www.microsoft.com/office/ork/2003/tools/default.htm*. You should probably download it to ensure that you're getting the most current version.

The two administrative files that you will work with are the following:

- OutlookSecurity.oft, the published Outlook form that you use to post messages in the public folder; in and of itself, this template doesn't provide any security.
- Hashctl.dll, the file for the Trusted Code control, a tool used by the template to specify which COM add-ins you trust to run within Outlook (remember, this only applies to Outlook 2002 and later versions).

To install the Outlook security package, run Admpack.exe. When you run Admpack, it installs the necessary files to a location you specify. However, you still have to use those files to implement the necessary security settings.

Installing the Trusted Code Control

Outlook allows you to write add-ins that are loaded into Outlook's process space; these add-ins use COM to communicate back and forth with Outlook. Because these add-ins run in-process, it's not a good idea to run add-ins from untrusted sources. When you install the security package, you gain the ability to create a list of trusted add-ins that clients can run without being prompted by Outlook security, provided you install the Trusted Code control. You only need this on your administrative machines; end users don't need it (and, in fact, shouldn't have it). Because the Hashctl.dll file is already installed on your machine, the only thing you really need to do is to register it, although Microsoft recommends moving the Hashctl.dll file to the \System32 folder in your Windows directory. To register the dynamic-link library (DLL), use this command:

```
regsvr32 hashctl.dll
```

Creating a Public Folder for Security Settings

Before you can create any settings, much less apply them, you'll need to create a top-level public folder to host the settings. The name of the folder is critical because Outlook is hardwired to look in that folder. You have three choices: if you create a folder named Outlook 10 Security Settings, Outlook 2003 (and Outlook 2002) uses the settings there, but Outlook 2000 won't see them. If you name the folder Outlook

Security Settings, Outlook 2000, Outlook 2002, and Outlook 2003 use the same settings. If you create both folders, each version uses its own folder (assuming that you've properly set the client-side registry key discussed later). Once you've created the folder (or folders), give all users Read access. Users who are allowed to change security settings should have permission to create, edit, and delete items in the folder.

Once you've created the folder, you're ready to publish the template and create an instance of the form. Do the following:

1. Start Outlook.
2. Open the OutlookSecurity.oft file. Outlook will prompt you for a location to save the new item you're creating from the template. The new item is never actually saved, so it doesn't matter which folder you choose.
3. While the template is open, use the Tools | Forms | Publish Form command. Give the form a meaningful name; if you're replacing the Outlook 2000–specific version of the form, give this form the same name as the old one. Make sure you specify the security settings public folder you created as the target location for the published form! Click Publish to publish the form.
4. Close the Default Security Settings template window. When Outlook asks if you want to save changes, make sure you do *not* do so.
5. Switch to the General tab of the public folder Properties dialog box, then use the When Posting To The Folder, Use drop-down list to specify the published form as the default.
6. After Outlook installs the template, open the public folder, then create a new item—name it Default Security Settings. You'll customize the default security settings by editing this item, and create new settings for different groups by creating copies of that default item or by creating new, blank entries and customizing them.

Filling Out the Template

The Outlook security form template has three tabs, each of which has its own set of controls that govern how Outlook's security features work, as follows:

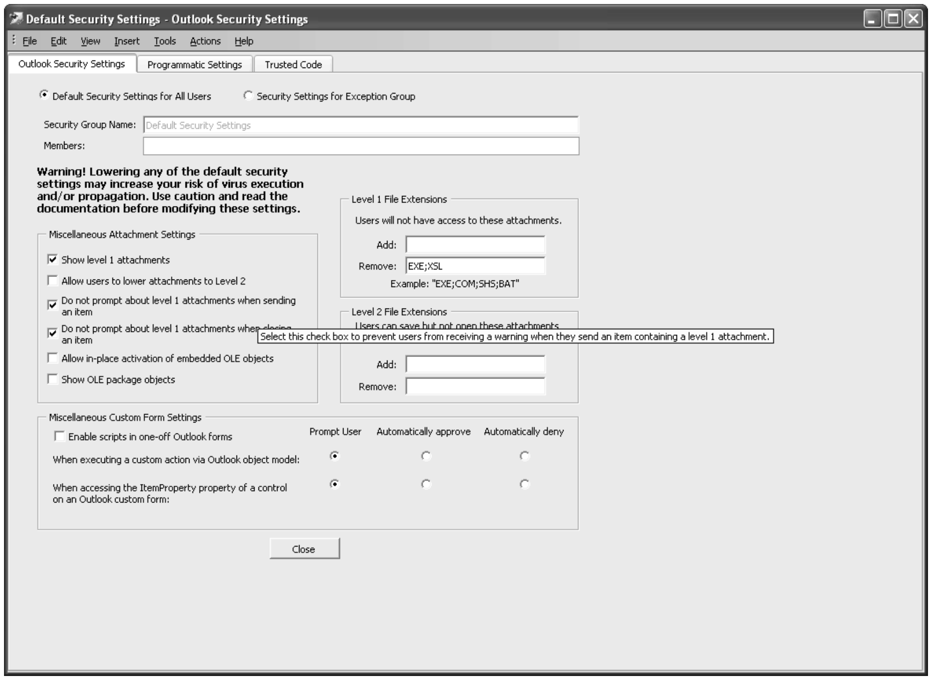
- The Outlook Security Settings tab controls general settings that are applied to Outlook clients.
- The Programmatic Settings tab controls what happens when outside applications try to use certain Outlook methods and properties, including sending mail, and retrieving address information.
- The Trusted Code tab lets you specify which Outlook COM add-ins you want to allow users to run without security prompts; note that these settings don't apply to other Office applications.

Note You cannot create or update the security settings template when you're using an Outlook profile that's set to use cached Exchange mode; Outlook will complain and tell you to switch to online mode when you attempt to save your updated form.

The Outlook Security Settings Tab

The Outlook Security Settings tab allows you to configure default settings that apply to all users. You can also apply settings to individual users or groups of users. The normal way to accomplish this is to create multiple post items using the form: one for the default settings you want to apply and one for each individual or group that needs special settings. When you use Microsoft Exchange 2000 Server or Exchange Server 2003, Outlook lets you specify Windows 2000 Server distribution groups to indicate who you want settings to apply to. If you're still hosting the folder on Microsoft Exchange 5.5, you must enter the name of each individual mailbox in the group, separated by semicolons, up to the hard-coded limit of 1000 names.

The tab itself is shown in Figure 13-2. The controls at the very top control who this copy of the form applies to. By default, the Default Security Settings For All Users option is selected, meaning that these settings apply to all Outlook users on this Exchange server. If you want to apply these settings to a subset of your users, select the Security Settings For Exception Group option, then fill in the Security Group Name and Members fields. Here's what the rest of the form's controls do:


Default Security Settings - Outlook Security Settings
File Edit View Insert Tools Actions Help
Outlook Security Settings
Programmatic Settings
Trusted Code
Close

[✓ Default Security Settings for All Users] Security Settings for Exception Group

Security Group Name: Default Security Settings

Members:

Warning! Lowering any of the default security settings may increase your risk of virus execution and/or propagation. Use caution and read the documentation before modifying these settings.

Miscellaneous Attachment Settings

- [✓] Show level 1 attachments
- [ ] Allow users to lower attachments to Level 2
- [✓] Do not prompt about level 1 attachments when sending an item
- [✓] Do not prompt about level 1 attachments when closing an item
- [ ] Allow in-place activation of embedded OLE objects
- [ ] Show OLE package objects

Select this check box to prevent users from receiving a warning when they send an item containing a level 1 attachment.

Level 1 File Extensions

Users will not have access to these attachments.

Add:

Remove: EXE;XSL

Example: "EXE;COM;SHS;BAT"

Level 2 File Extensions

Users can save but not open these attachments.

Add:

Remove:

Miscellaneous Custom Form Settings

- [ ] Enable scripts in one-off Outlook forms

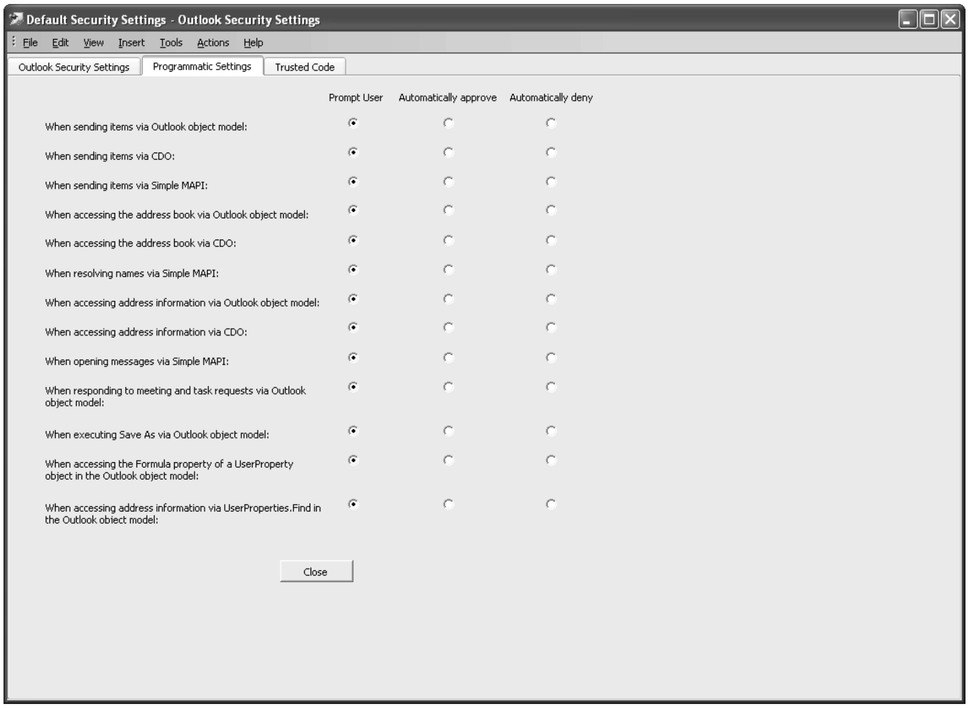
	Prompt User	Automatically approve	Automatically deny
When executing a custom action via Outlook object model:	●	○	○
When accessing the ItemProperty property of a control on an Outlook custom form:	●	○	○

Figure 13-2. *The Outlook Security Settings tab gives you control over how Outlook handles attachments.*

- The Level 1 File Extensions and Level 2 File Extensions control groups let you specify which file types are included in each group. You can add or remove attachment types by putting the extensions in the appropriate fields and saving the items; unfortunately, there's no way to see the current list of Level 1 or Level 2 extensions.
- The Miscellaneous Attachment Settings controls give you control over Outlook behavior when it encounters a Level 1 item.
  - The Show Level 1 Attachments check box tells Outlook whether or not you want the names of Level 1 attachments to be visible in the InfoBar, even though users might not be able to get them.
  - The Do Not Prompt About Level 1 Attachments When Sending An Item and Do Not Prompt About Level 1 Attachments When Closing An Item check boxes govern whether Outlook warns you when you include a forbidden attachment in a message you're composing.
  - The Allow In-Place Activation Of Embedded OLE Objects check box controls whether Outlook allows in-place activation or not. In-place activation actually turns on an embedded application, making its menus and toolbars visible and active within the multiple-document interface (MDI) frame of an existing application. Because this actually cedes control of the current application to the automation server for the embedded object, it poses a small security risk, so Microsoft turns it off by default. This option lets you turn it on.
  - The Show OLE Package Objects check box controls whether Outlook shows OLE binder objects; again, this represents a small security risk.
- The Miscellaneous Custom Form Settings control group includes some oddball options that don't really fit anywhere else. The Enable Scripts In One-Off Outlook Forms check box controls whether individual unpublished Outlook forms (that is, those distributed as .oft files or by using the Send Form Definition With Item check box when the item is sent) can run scripts or not; the remaining options regulate what happens when external code, macros, forms, or COM add-ins attempt to use the Outlook object model to execute forms' custom actions or access the properties of a form control that specifies what Outlook item it's bound to. You have three options:
  - The Prompt User option causes Outlook to display a dialog box prompting the user to choose whether to allow access or not.
  - The Automatically Approve option tells Outlook to allow all access without displaying a warning.
  - The Automatically Deny option tells Outlook to deny all requests without asking.

The Programmatic Settings Tab

The Programmatic Settings tab, shown in Figure 13-3, features an impressive set of options. Fortunately, these break down into a fairly simple taxonomy; you use this form to specify which actions various types of code can take, according both to the requested action and the type of interface used to make the request. Applications, macros, add-ins, and other executables can use three different interfaces to request services from Outlook, as follows:


Default Security Settings - Outlook Security Settings
File Edit View Insert Tools Actions Help
Outlook Security Settings
Programmatic Settings
Trusted Code
Close

	Prompt User	Automatically approve	Automatically deny
When sending items via Outlook object model:	●	○	○
When sending items via CDO:	●	○	○
When sending items via Simple MAPI:	●	○	○
When accessing the address book via Outlook object model:	●	○	○
When accessing the address book via CDO:	●	○	○
When resolving names via Simple MAPI:	●	○	○
When accessing address information via Outlook object model:	●	○	○
When accessing address information via CDO:	●	○	○
When opening messages via Simple MAPI:	●	○	○
When responding to meeting and task requests via Outlook object model:	●	○	○
When executing Save As via Outlook object model:	●	○	○
When accessing the Formula property of a UserProperty object in the Outlook object model:	●	○	○
When accessing address information via UserProperties.Find in the Outlook object model:	●	○	○

Figure 13-3. *Control programs' access to the Outlook object model and address book with the Programmatic Settings tab.*

- **Outlook object model** The Outlook object model allows you to manipulate data stored in Outlook folders using a set of built-in interfaces that work with any language that supports COM; most of the time, people leverage this object model with code written in Visual Basic (VB) or Microsoft Visual Basic for Applications (VBA) to do this.
- **Simple MAPI** MAPI comes in two varieties: Simple and Extended. Simple MAPI enables developers to add basic messaging functionality, such as sending and receiving messages, to their Windows-based applications. Extended MAPI allows applications more control; it's also the only way an external application can use Outlook's data without triggering the Outlook object model guards.

- **CDO** CDO provides messaging and collaboration functionality; CDO lets you do things like schedule appointments, look up contacts, and perform other neat tricks. CDO 1.21 is actually a client-side COM wrapper for the MAPI library; any language that can use COM can use CDO. CDO implements most—but not all—MAPI functionality (but more than Simple MAPI). Don't confuse this CDO with the CDO for Exchange Management (CDOEXM) or CDOSYS libraries shipped as part of Exchange and Windows—same name, but different functionality.

Each potential action (sending items using CDO, looking up address book items with Simple MAPI, and so on) is independently controlled. When any third-party software on the client attempts to do something, the behavior you specify takes effect: the client either approves the request automatically (which is what happens in Outlook 2000 without the security update), automatically denies it, or prompts the user (the default behavior in Outlook with the security update).

The Trusted Code Tab

The Trusted Code tab is visually uninteresting, so I don't show it here. It contains a list box and two buttons: Add and Remove. You use the buttons to add or remove COM add-ins to the list; Outlook clients trust (that is, allow to run) any COM Outlook add-in on the list. However, remember that specifying an add-in on this tab doesn't make it run, nor does it install it on the client. Specifying an add-in here just means that the add-in has permission to call the Outlook object model without triggering the object model guards; add-ins that call CDO are still subject to the guards.

There are a few nuances to using this tab. First of all, you can only specify Outlook add-ins, not, for example, a Microsoft Word add-in that happens to call Outlook object model functions. Second, if you later replace a trusted add-in with a newer version, you'll need to remove the old version from the list and add the new one, even if they have the same file name.

Deploying Outlook Security Settings

After you configure the settings you want by creating items on the Exchange server, you still have to force Outlook to use those settings. To enable this behavior, you'll need to deploy a new registry key to the client computers; that's why this is best done during your initial rollout of Office or Outlook.

The simplest way to do this is to use the Custom Installation Wizard to include the registry key in a transform when you deploy the Office System. If you've already deployed Office, you can use the Custom Maintenance Wizard to add the registry key information to the client. However, neither of these methods is enforced, so clients can manually change their local settings. If you want the new registry key to be enforced, you'll need to deploy it with a system or group policy.

If you're managing your Office installation with policies, adding this behavior is simple. Just add the correct policy template (.adm file) so that your policy object includes the necessary key, then set the policy to apply to the target users. If you use the System Policy Editor provided with the Office Resource Kit Toolbox, the correct templates are already loaded. If you use the Active Directory Group Policy Object snap-in, you'll need to add the templates manually. The policy file automatically passes your customized security settings to client computers each time users log on to the system.

So, what's the magic key you have to modify? The registry value is a DWORD named *CheckAdminSettings* and located under H*KEY_CURRENT_USER\Software\Policies\Microsoft\Security\*. The value you put here determines where Outlook searches for security settings. Table 13-2 shows which values do what.

Note You can't save this registry key to a file and mail it out because .reg files are blocked by the security update, and they're also blocked by the default Outlook settings that apply when no security settings are present on the server. Besides, the best way to ensure that users don't change these settings is to force their application using Group Policy Objects (GPOs), so that's what Microsoft recommends (although you can certainly add a .reg file that runs as part of a user logon script as an alternative).

Table 13-2. CheckAdminSettings Values

Value	What Outlook 2003 Does
Key not present	Uses its default settings.
0	Uses its default settings.
1	Looks for settings in the Outlook Security Settings folder, applying them according to the defaults and specific users you've specified.
2	For Outlook 2002 and Outlook 2003 only: Looks for settings in the Outlook 10 Security Settings folder, ignoring any settings in the Outlook Security Settings folder. Use this value when you want Outlook 2002 or Outlook 2003 and Outlook 2000 to use different settings.
Anything else	Uses its default settings.

Customizing Outlook Security Settings for End Users

You might choose not to deploy the public folder that applies settings to your Outlook clients (although by not doing so you're skipping a valuable security feature). If you don't, then Outlook 2003 will still apply the Level 1 and Level 2 restrictions discussed earlier, but with a twist: each user can customize his or her own copy of Outlook to control the Level 1 and Level 2 lists. The trick is to add a new string value named Level1Remove to the *HKEY_CURRENT_USER\Software\Microsoft\Office\11.0\Outlook\Security* key. The extensions you add here (separated by semicolons if there's more than one) are removed from the list of blocked Level 1 attachments, so creating a value of exe; pl would allow executables and Perl scripts to be saved to disk instead of blocking them completely. Actually, the extensions you specified are demoted from Level 1 to Level 2; they're not unblocked completely. End users cannot demote file types from Level 2 to being unprotected; only administrators can do so.

If you want to add a new file type to the Level 1 list, you can do so by creating a new string value named Level1Add beneath the *HKEY_CURRENT_USER\ Software \Microsoft\Office\10.0\Outlook\Security* key.

Tip Sue Mosher maintains a page that includes links to tools that your users can use to customize their local attachment settings without directly editing the registry. See *http://www.slipstick.com/outlook/esecup/getexe.htm*. Alternatively, you can always set a value for Level1Remove as part of a GPO or system policy; that way, users get the values you want without having to spend time fiddling with their local settings.

Note To check whether a user has customized his or her Outlook security settings, use the Help | About Microsoft Outlook command. Above the license information, Outlook displays the security mode (mine says Security Mode: Default); a user-customized machine will say Security Mode: User Controlled.

Of course, it is more likely that you'll want to *prevent* users from customizing their own security settings. The easiest way to do this is to add a new REG_DWORD value named *DisallowAttachmentCustomization* to the Outlook key at *HKCU \Software\Policies\Microsoft\Office\11.0\Outlook*. When this value is present, Outlook will ignore the Level1Add and Level1Remove keys mentioned earlier.

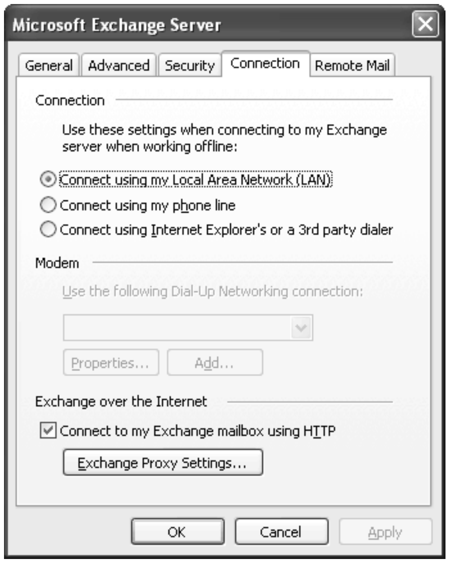
Setting Up RPC over HTTP

As you found in Chapter 11, most of the work for setting up RPC over HTTP actually has to be done on the server side. On the client side, you'll need to ensure that your servers have Microsoft Windows XP. If you're using Service Pack 1, you'll need the Q331320 hotfix, which is included with Service Pack 2 and later. You'll also need to have Exchange Server 2003 running on Windows Server 2003 for the front-end and back-end servers your users communicate with, and all global catalogs and domain controllers that your servers and clients talk to must also be running Windows Server 2003.

Note The Office Resource Kit contains a wealth of material on deploying RPC over HTTP using the Custom Installation Wizard, which makes it possible to seamlessly enable it for some or all of your users at the time you deploy it. See *http://www.microsoft.com/office/ork/2003/three/ch8/outc07.htm* for more details.

The settings for RPC over HTTP are associated with individual profiles and can only be applied to a single Exchange server account in each profile. You modify these settings using the same interface you're probably familiar with, but the settings themselves are different. (Remember, you must already have set up your Exchange servers and global catalogs as described in Chapter 11.)

The key to getting RPC over HTTP set up for Outlook is found in a single simple check box, Connect To My Exchange Mailbox Using HTTP, shown in Figure 13-4. (You get to this check box by editing an account with the Tools | E-Mail Accounts command, clicking Change, clicking More Settings, and clicking the Connection tab.) This check box is visible when you're running Outlook 2003 on a system that meets the prerequisites and talking to an Exchange server that meets its prerequisite requirements. If any component is missing or misconfigured, the check box won't appear.



Connection

Use these settings when connecting to my Exchange server when working offline:

(●) Connect using my Local Area Network (LAN)
( ) Connect using my phone line
( ) Connect using Internet Explorer's or a 3rd party dialer

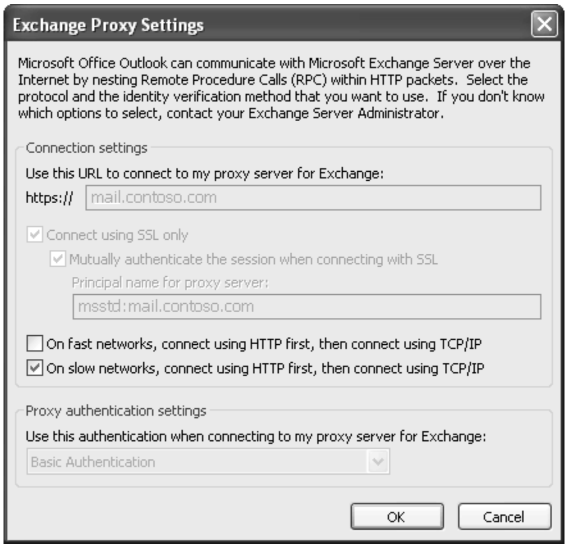
Use the following Dial-Up Networking connection:

Exchange over the Internet

[✓] Connect to my Exchange mailbox using HTTP

Figure 13-4. *The Connection tab has the key check box for enabling RPC over HTTP*

After you select the check box, of course, the real fun begins. The Exchange Proxy Settings button controls the appearance of the Exchange Proxy Settings dialog box (see Figure 13-5). You can specify the URL for your Exchange server (which, for a standard Exchange Server 2003 installation, will be the same as the name of the front-end server) and whether you want to require the use of SSL. For maximum security, you should ensure that the Connect Using SSL Only and Mutually Authenticate The Session When Connecting With SSL check boxes are both selected; this combination provides the best protection against spoofing and eavesdropping. The other settings are pretty much irrelevant from a security standpoint, with the exception of the Use This Authentication When Connecting To My Proxy Server For Exchange control.



Microsoft Office Outlook can communicate with Microsoft Exchange Server over the Internet by nesting Remote Procedure Calls (RPC) within HTTP packets. Select the protocol and the identity verification method that you want to use. If you don't know which options to select, contact your Exchange Server Administrator.

Connection settings

Use this URL to connect to my proxy server for Exchange:

https:// mail.contoso.com

[✓] Connect using SSL only

[✓] Mutually authenticate the session when connecting with SSL

Principal name for proxy server: msstd:mail.contoso.com

[ ] On fast networks, connect using HTTP first, then connect using TCP/IP

[✓] On slow networks, connect using HTTP first, then connect using TCP/IP

Proxy authentication settings

Use this authentication when connecting to my proxy server for Exchange: Basic Authentication

Figure 13-5. *The Exchange Proxy Settings dialog box.*

Other Nifty Tricks

There are two other useful things to know about Outlook 2003 RPC over HTTPS support. The first is that you can disable the user interface controls that let users change RPC over HTTPS behavior. This is useful if you want to ensure that your users don't set it up on their own, or if you want to prevent them from changing settings once you've deployed them. To do this, add the EnableRPCTunnelingUI value (a REG_DWORD) to *HKEY_CURRENT_USER\Software\Policies\Microsoft\Office\11.0\Outlook\RPC*. When this value is set to 0, the user interface (UI) controls are hidden; when it's set to 1, or not present, the UI controls are visible as long as Outlook is running on a machine that meets the operating system requirements.

The other useful thing to know is that you can turn on RPC over HTTPS at a later date, after your initial Outlook 2003 deployment. To do this, you should use the Office Resource Kit's Custom Maintenance Wizard, which lets you make some types of configuration changes and deploy them as files that can automatically update installed Office configurations. To learn more about the Custom Maintenance Wizard, see *http://www.microsoft.com/office/ork/2003/three/ch10/OutE01.htm*.

Using S/MIME

S/MIME can be a little daunting for end users, because it relies heavily on algorithms and protocols that can charitably be described as complex. Microsoft has done its best to simplify the interface and behavior in Outlook while still giving power users lots of knobs to adjust.

Managing Certificates

Before you can do much of anything with S/MIME, you'll need an X.509v3 certificate that is flagged for use with e-mail (either as a signing certificate, an encryption certificate, or a dual-use certificate). Accordingly, certificate management is the logical place to start.

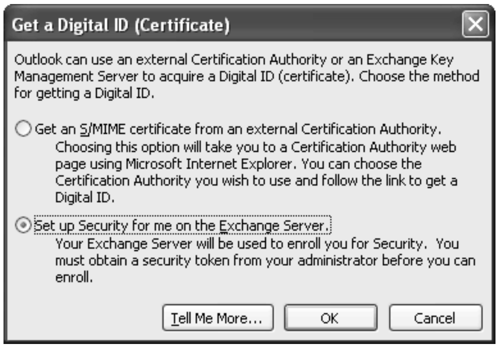
Getting a New Certificate

Outlook happily uses any properly constructed X.509v3 certificate for which you have the private key, provided you can get the certificate into your local user profile on the machine you are currently using. Depending on whether you're setting up a complete (or outsourced) public-key infrastructure (PKI) or whether individual users want certificates, you can enroll users through the Exchange 2000 Server Key Management Service (KMS), manually or automatically enroll through a Windows Server 2003 CA, or manually request individual certificates from a third-party CA.

If you are not using a Windows Server 2003 CA to automatically or manually enroll a certificate, you tell Outlook to request your new certificate by launching Outlook, choosing the Tools | Options command, clicking the Security tab, and clicking Get A Digital ID. What happens next depends on whether you're requesting a certificate through the Exchange KMS or from an external CA. For more information on how to set up and utilize certificate auto-enrollment with Windows Server 2003, refer to the white paper at *http://www.microsoft.com/technet/prodtechnol/windowsserver2003/plan/autoenro.asp*.

Getting a Certificate from an Internal Certificate Server The most common way for enterprise users to get a certificate is through the Windows Server 2003 Certificate Services component, which (as you read in Chapter 12, "Secure E-Mail") takes the place of the Exchange 2000 Server KMS. The preferred deployment model now in a Windows XP and Windows Server 2003 environment is to use user certificate auto-enrollment, but support for Exchange KMS advanced security using the Exchange tools included in the Active Directory Users and Computers snap-in is still available. Although the new model is easiest with little or no user interaction, from the legacy Outlook and KMS side, the manual process is still simple; it begins when clients use Outlook to generate a temporary key, which is then used to protect the initial client-to-CA exchange. You can provide the token in an e-mail message or using another (presumably more secure) offline channel; either way, the user must have the token before proceeding.

After you click Get A Digital ID, you'll see the dialog box shown in Figure 13-6. This dialog box only appears for accounts that have been enrolled in Advanced Security; if you don't see it, the logged-in profile's account needs to be enrolled before proceeding. Because you want a certificate issued by the local Exchange organization's certificate server, choose the Set Up Security For Me On The Exchange Server option and click OK.



Get a Digital ID (Certificate)

Outlook can use an external Certification Authority or an Exchange Key Management Server to acquire a Digital ID (certificate). Choose the method for getting a Digital ID.

○ Get an S/MIME certificate from an external Certification Authority.
Choosing this option will take you to a Certification Authority web page using Microsoft Internet Explorer. You can choose the Certification Authority you wish to use and follow the link to get a Digital ID.

[✓ Set up Security for me on the Exchange Server.]
Your Exchange Server will be used to enroll you for Security. You must obtain a security token from your administrator before you can enroll.

Figure 13-6. *Pick the certificate source you want to use for your request.*

Outlook asks you to pick a name for this digital ID and enter your token, then you are asked to assign a password. Note that this password is completely separate from your Windows account password—and it should be different, so that compromising one doesn't affect the other. At this point, your request is sent to the certificate server, which approves or rejects it (usually fairly quickly). First, though, you have to close the Outlook Options dialog box; you will eventually receive a signed status message from the CA indicating either that your enrollment was rejected or accepted. For requests that are accepted, you'll be prompted to enter your password so that Outlook can publish your certificate in the Global Address List (GAL) for you (you can do so manually, too, as you'll see later).

Using an External CA Individual users can always request their own certificates from an external CA and install them in Outlook. Presuming that you choose a widely known CA, this might be an attractive alternative to setting up your own PKI, particularly for small numbers of users. Certificates are available from various sources. For example, Thawte offers a free end-user certificate that's adequate for many individuals' uses; various professional, social, and government organizations either are or will soon be issuing certificates to their members.

If you click Get An S/MIME Certificate From An External Certification Authority, Outlook takes you to a page associated with an external CA. By default, the page you get is one hosted at the Microsoft Office Assistance Center (currently *http://office.microsoft.com/marketplace/PortalProviderPreview.aspx?AssetID=EY010504841033*) that lists third-party CAs that you can use. The CAs listed there offer personal certificates; each one has slightly different enrollment and certification procedures. However, using the policy controls I discuss subsequently, you can send your organization's Outlook users to whatever CA you prefer.

Once you've enrolled with the remote CA and received your certificate, you'll need to import it, as discussed in the section "Importing and Exporting Certificates" later in the chapter.

Publishing Your Certificate in the GAL

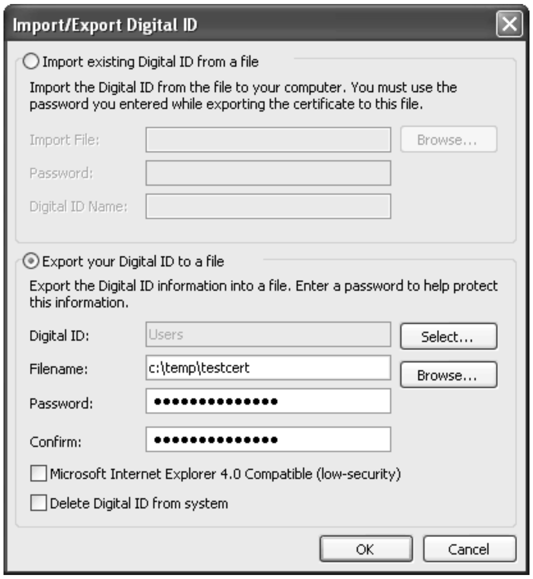
If you're using an enterprise CA, newly issued certificates are published in the GAL. However, users who go out and get their own certificates will find that their certificates aren't in the GAL. If Alice and Bob want to exchange encrypted messages, and neither has a certificate in the GAL, Alice must first send Bob a signed message. When Bob opens it, his Outlook client adds Alice's certificate to the local store, at which point he can use it to send her an encrypted reply. This requirement is a hassle, so Outlook provides a way for you to publish arbitrary certificates that you control as attributes for your user account; this, in turn, makes them visible in the GAL. The steps required to publish your certificate are simple:

1. Launch Outlook while logged in with the account you want the certificate associated with. (Of course, you have to have access to the desired certificate in some form, either in the local machine certificate store or on a smart card or other portable token.)
2. Use the Tools | Options command to open the Options dialog box, then click the Security tab.
3. Click Publish To GAL. Outlook warns you that it's about to publish your default certificate (the one identified in the Default Setting field of the Secure E-Mail command group, which is described in the next section) to the GAL; click OK if you want it to do so.

Importing and Exporting Certificates

Once you have a certificate, it's not necessarily tied to a single machine. CryptoAPI allows whoever generates the key to specify whether a key can be exported or not; by default, most certificates generated by the Windows Certificate Service or third-party CAs come from private keys that are marked as exportable. As long as the private key is exportable, you can use Outlook to export the key pair and associated certificate to a file, which can then be imported on another machine. For example, I can receive a certificate on my desktop machine, then export it and import it on my laptop so I have access to encrypted mail (and other resources) from both machines.

You export and import certificates from Outlook using the Import/Export button on the Security tab of Outlook's Options dialog box. Clicking that button produces the Import/Export Digital ID dialog box shown in Figure 13-7. Use the two options to select whether you want to import or export; the associated controls in each group let you specify where the file containing the certificate is located and what the associated password is. The Delete Digital ID From System check box is worth special mention: when selected, after exporting your private key to the file Outlook removes it. Use this option when you want to migrate a certificate from one machine to another; leave it cleared if you want to copy the key material.


Import/Export Digital ID

( ) Import existing Digital ID from a file

Import the Digital ID from the file to your computer. You must use the password you entered while exporting the certificate to this file.

Import File:
Password:
Digital ID Name:

(●) Export your Digital ID to a file

Export the Digital ID information into a file. Enter a password to help protect this information.

Digital ID: Users
Filename: c:\temp\testcert
Password: ••••••••••••••
Confirm: ••••••••••••••

☐ Microsoft Internet Explorer 4.0 Compatible (low-security)

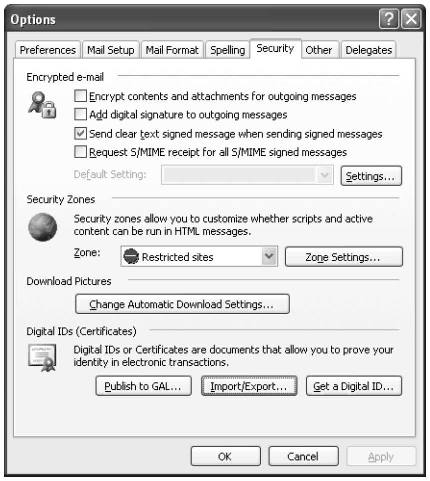
☐ Delete Digital ID from system

Figure 13-7. *You can use Outlook's import/export feature to move or copy your certificates between machines, but be careful not to unnecessarily expose them to compromise.*

Note You can also import an existing certificate from your own or a third-party CA; as long as it has the flags indicating that it's usable for e-mail encryption and signatures, you'll be able to use it with Outlook.

Setting S/MIME Options

Figure 13-8 shows the Security tab of Outlook's Options dialog box. The Encrypted E-Mail control group on this tab allows you to set the defaults you want Outlook to use for S/MIME traffic. You can choose to sign, encrypt, or sign and encrypt outbound messages by selecting the appropriate check boxes. In addition, you can choose whether signed messages should be clear-signed or opaque-signed and whether or not you want your messages to include requests for signed return receipts.



Encrypted e-mail

- [ ] Encrypt contents and attachments for outgoing messages
- [ ] Add digital signature to outgoing messages
- [x] Send clear text signed message when sending signed messages
- [ ] Request S/MIME receipt for all S/MIME signed messages

Default Setting:

Security Zones

Security zones allow you to customize whether scripts and active content can be run in HTML messages.

Zone: Restricted sites

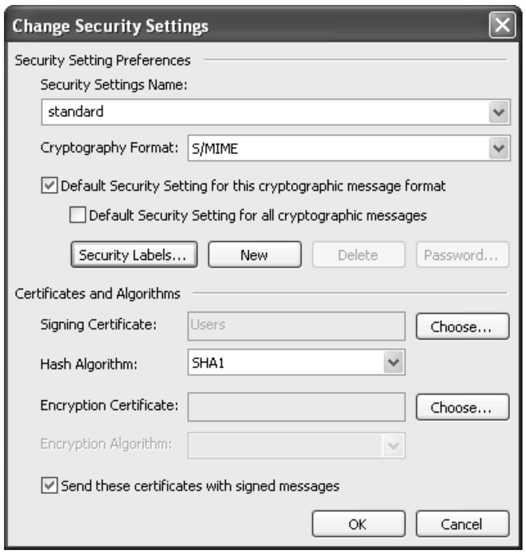
Download Pictures

Digital IDs (Certificates)

Digital IDs or Certificates are documents that allow you to prove your identity in electronic transactions.

Figure 13-8. *Use the Encrypted E-Mail control group on the Options dialog box Security tab to control Outlook's S/MIME behavior.*

The most interesting control in this group is the Default Setting drop-down list and the associated Settings button; that's because these settings control the algorithms and message format you use when sending secure mail. When you click Settings, the Change Security Settings dialog box, shown in Figure 13-9, opens. Each security settings object contains your preferences for the certificate you want to use for signing and encrypting messages and the algorithms you prefer for each use. The controls in the dialog box are self-explanatory, so instead of reiterating what they do, it's more useful for me to explain why they're there in the first place.



Security Setting Preferences

Security Settings Name: standard

Cryptography Format: S/MIME

- [x] Default Security Setting for this cryptographic message format
- [ ] Default Security Setting for all cryptographic messages

Certificates and Algorithms

Signing Certificate: Users

Hash Algorithm: SHA1

Encryption Certificate:

Encryption Algorithm:

- [x] Send these certificates with signed messages

Figure 13-9. *Create groups of security settings for use with different certificates or recipients.*

Remember that a certificate is just a credential. We all carry around multiple credentials: my driver's license isn't useful at the video store, and my bank card isn't useful when I want to board an airplane—each credential has its own purpose and set of attributes. Likewise, it's increasingly common for organizations that deploy PKIs to issue separate certificates for different purposes: every user might get one for signing e-mail, but only the legal and merger departments might need one for encryption, and only the IT department gets certificates that can be used to sign macros or Office objects. This partitioning means that it might be useful to specify different algorithms or certificates for signatures and encryption, or even to maintain different "work" and "home" settings for users with business and personal certificates. That's one reason Outlook supports multiple sets of credentials, the other being its support for security labels (part of the DMS support included in Outlook).

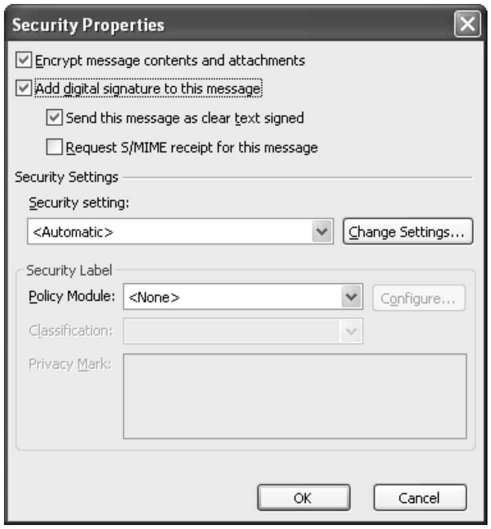
Signing or Encrypting a Message

Actually signing or encrypting messages is easy with Outlook. You can always sign or encrypt individual messages, either using the Outlook toolbar or by opening the message properties; you can also tell Outlook to sign or encrypt all messages by default.

Encrypting or Signing a Single Message

To secure a message as you're composing it in Outlook 2003, click the Options button in the toolbar; when the Message Options dialog box appears, click the Security Settings button and you'll see the Security Properties dialog box, shown in Figure 13-10. The Encrypt Message Contents And Attachments and Add Digital Signature To This Message check boxes do just what you'd expect; you can select either or both of them to add protection to your message when it's sent. When signing a message, you have two additional options: you can clear-sign the message (so that non-S/MIME-aware mail software can display the message contents), and you can request a signed return receipt—this proves that the recipient opened the message at a particular date and time, because it's signed with the recipient's private key.

Caution When you're connected to your Exchange server, as soon as you add an attachment to a message in Outlook, it begins uploading the attachment in the background to the server. That conversation isn't encrypted (unless you've turned on RPC encryption, discussed later in the chapter), so be aware that an adversary might see some of your attachment if he or she can sniff traffic on the network.


Security Properties

☑ Encrypt message contents and attachments
☑ Add digital signature to this message
☑ Send this message as clear text signed
☐ Request S/MIME receipt for this message

Security Settings
Security setting: <Automatic>

Security Label
Policy Module: <None>
Classification:
Privacy Mark:

Figure 13-10. *To sign or encrypt a message, just select the check boxes that correspond to the desired security features.*

You can also use this dialog box to choose a set of S/MIME parameters for the message; your choices are taken from the sets of S/MIME settings you defined on the Security tab of the Options dialog box, as discussed in the preceding section. You can also set a security label from this dialog box, but you won't be able to do so unless you're using a policy module, and there aren't any widely available ones except those used with DMS.

Tip If you want to use security labeling, you can implement your own policy module; see *http://msdn.microsoft.com/library/en-us/dnout2k2/html/odc_olseclabelapi.asp* for details.

Updating the Outlook Toolbar

Outlook's Standard toolbar includes toolbar buttons for encrypting and signing the current message, but they're not visible by default. Fortunately, you can customize the toolbar to include the icons—just use the Tools | Customize command, which displays the Customize dialog box. Here's what to do:

1. Open a message (new or old, it doesn't matter). If the toolbar you want to put the buttons on isn't visible, make it visible with the View | Toolbar submenu.

2. Choose the Tools | Customize command. When the Customize dialog box opens, click the Commands tab.
3. Select Standard in the Categories list. The Commands pane lists the available toolbar buttons. Near the bottom of the Commands list, you'll see two icons: Encrypt Message Contents And Attachments and Digitally Sign Message.
4. Drag the icons to the desired positions on the toolbar.
5. Click Close.

Applying Policy Controls for S/MIME Use

It's all well and good to provide users a way to create signed and encrypted messages when they want to, but in many environments it's necessary to force or restrict a particular set of security settings. One little-noticed change in Outlook 2003 is that it incorporates a broad set of policy settings that allow centralized administrative control of encryption and security behaviors. These settings are available through Group Policy templates; the settings described in Table 13-3 are applied under the *HKEY_CURRENT_USER\Software\Policies\Microsoft \Office\10.0\Outlook\Security* key.

Table 13-3. Policy Settings for Outlook Cryptographic Features

Value Name	Data Type	Supported Values	Notes
AlwaysEncrypt	DWORD	1: All outgoing messages are encrypted. 0: Default; outbound messages are only encrypted by user request.	If Outlook cannot find a certificate for a recipient, you'll see an error dialog box like the one shown in Figure 13-11.
AlwaysSign	DWORD	1: All outgoing messages are signed. 0: Default; outbound messages are only signed by user request.	
ClearSign	DWORD	1: All outbound messages include the message in cleartext, with a separate signature part. 0: Messages are signed with a combined signature/message block; the message is not present in cleartext.	Clear-signed messages allow recipients who aren't using S/MIME-capable clients to read the message, even though they cannot verify the signature.

Table 13-3. Policy Settings for Outlook Cryptographic Features *(continued)*

Value Name	Data Type	Supported Values	Notes
RequestSecureReceipt	DWORD	1: All outbound messages contain a signed receipt request; compliant clients will automatically return signed receipts when the message is opened. 0: Receipts must be individually requested.	S/MIME receipts are digitally signed, as is the request. The S/MIME specification requires compliant clients to honor receipt requests, without including an option to suppress receipts like Outlook does for ordinary messages.
ForceSecurityLabel	DWORD	1: Every outbound message must be given a security label before it's sent. 0: Unlabeled messages can be sent.	This option is only useful if you're using a security labeling policy module.
ForceSecurityLabelX	Binary	All messages sent automatically include this administrator-defined label.	Label must be stored as an ASN.1-encoded string. If you don't know what that means, you probably won't benefit from this feature.
SigStatusNoCRL	DWORD	1: If the certificate revocation list (CRL) for a signing certificate cannot be found, treat it as an error. 0: Treat the missing CRL as a warning.	This option gives you control over whether users can accept messages with missing CRLs as valid or not.
SigStatusNoTrustDecision	DWORD	2: "No trust" decisions are treated as errors. 1: "No trust" decisions are treated as warnings. 0: "No trust" decisions are allowed.	This value controls what Outlook does when the signatures and certificates on a message don't chain upward to a trusted root. Setting this value to zero can lead to malicious spoofing if you are not careful. See the Microsoft white paper at *http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/WinXPPro/support/tshtcrl.asp*.

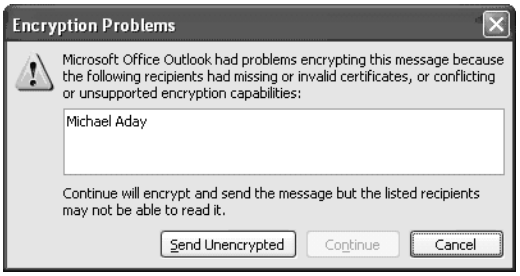
(continued)

Table 13-3. Policy Settings for Outlook Cryptographic Features *(continued)*

Value Name	Data Type	Supported Values	Notes
PromoteErrorsAsWarnings	DWORD	1: Treat level 2 errors as warnings 0: Treat level 2 errors as errors	Level 2 errors occur when the message signature appears to be valid, but something else is wrong (for example, no CRL can be found, the CRL or certificate trust list [CTL] is out of date, or the issuer certificate is missing).
PublishToGALDisabled	DWORD	1: Disables the Publish To GAL button. 0: Button functions normally.	
FIPSMode	DWORD	1: Force Outlook to FIPS 140-1 compliance mode. 0: Outlook is not forced to FIPS 140-1 mode.	As described in Chapter 2, FIPS 140 is the U.S. government standard for cryptographic algorithm performance and security. When Outlook is in FIPS 140-1 mode, the set of signing and encryption algorithms it can use is somewhat restricted.
WarnAboutInvalid	DWORD	2: Never give users warnings when invalid or improper certificates or signatures are found. 1: Always show the Secure E-Mail Problem dialog box when a problem occurs. 0: Ask the user each time whether a particular warning should be displayed.	By default, Outlook will warn users when a problem occurs in a secured message, along with displaying a Show And Ask check box that lets users ignore the problem and see the message anyway. This value controls whether users are given that choice when a problem message is encountered.
DisableContinueEncryption	DWORD	1: Hide the Continue Encryption button. 0: Show the button.	When you send a message to multiple recipients, if for some reason you can't send to a particular recipient, Outlook displays a button that lets you continue encrypting the message to the remaining recipients. If you want to prevent users from sending unencrypted messages, you'll have to turn this off.

Table 13-3. Policy Settings for Outlook Cryptographic Features *(continued)*

Value Name	Data Type	Supported Values	Notes
RespondToReceiptRequest	DWORD	0: Always send signed receipts when requested; only prompt for a password if it's needed. 1: Always send signed receipts; always prompt for the password. 2: Never send signed receipts. 3: Force receipts to be sent.	Because signed receipts have to be signed, you'll have to enter your credentials to sign the receipt. If you cancel instead, the receipt won't be sent. Tweaking this value lets you control whether the receipt is sent or not.
NeedEncryptionString	REG_SZ	String that is displayed when the user tries to open an encrypted message for which he or she doesn't have an appropriate certificate.	This message can be used to tell users where or how to get their certificates.
Options	DWORD	1: Don't warn users when they try to open a signed message with an invalid signature. 0: Warn users.	
MinEncKey	DWORD	Set to the minimum key length to be used when sending encrypted messages.	Set this value to the key length you want to use: 40, 64, 128, or 168 bits. (Note that 40- and 64-bit encryption are too weak for most commercial applications.)
requiredCA	REG_SZ	Gives the name of the required CA; when this is set, Outlook won't sign or encrypt mail with certificates issued by any other CA.	
EnrollPageURL	REG_SZ	URL for the page that users should be sent to when they click Get A Digital ID.	This string can contain three expansion variables: %1 is replaced with the user's display name, %2 is replaced with the user's SMTP address, and %3 is the code page ID for the user's preferred language.


Encryption Problems

Microsoft Office Outlook had problems encrypting this message because the following recipients had missing or invalid certificates, or conflicting or unsupported encryption capabilities:

Michael Aday

Continue will encrypt and send the message but the listed recipients may not be able to read it.

Figure 13-11. *Outlook complains if it cannot find a recipient certificate.*

Using Information Rights Management

The IRM features of Outlook give senders more control over their e-mail by allowing them to specify that a message cannot be copied, forwarded, printed, or used past a certain date. It's important to point out that this protection is not absolute: a clever recipient can always use a digital camera to snap a quick picture of the message on screen; failing that, a pencil and paper allow even technophobes to accurately capture message content. The point of IRM, though, is to make accidental misuse of content less likely and to provide some degree of protection against purposeful misuse, and for those purposes it's successful.

To use IRM, your users will need a server running Windows Server 2003 and Windows RMS set up inside your corporate firewall. Microsoft has taken the wise step of making an RMS server available to anyone with a Microsoft Passport account. This service allows use of RMS with some caveats, the biggest being that it's a free, trial, unsupported service. It's a good way to experiment with RMS features, though; it's likely that Microsoft will extend this into some kind of paid service for people who want RMS functionality without the overhead of maintaining their own RMS locally. In this chapter, I describe using RMS with both corporate and Passport-based credentials; for more information on setting up RMS on corporate networks, see Chapter 12. However, I limit my discussion to the process of setting up and using RMS in Outlook. The steps required to use it in Microsoft Excel, Word, and Microsoft PowerPoint are similar.

Setting Up for IRM

When you install Office, you get a version of the IRM client. The first time you try to use the IRM feature of Office, you might be prompted to download an updated version of the Windows RMS client. There are actually two separate applications that use the Windows RMS client in an IRM deployment: the Office System, and the Rights Management add-on for Internet Explorer to enable users without the Office System to view RMS-protected content. Either way, the client installation is very straightforward, so I don't cover it here: Office prompts you to download a single Windows installer (.msi) file, and it doesn't ask you to do anything except accept the end-user license agreement.

Once you've installed the client, the first time you try to use an IRM feature, you'll be prompted to establish a set of credentials. This process is fairly simple; the Service Sign-Up Wizard leads you through each step. The process begins with a page that explains that the trial service requires a Microsoft Passport account, that Microsoft won't access your data unless a court forces them to, and that the service might be discontinued but that you'll get a warning first. Next, you'll be asked whether you have a Passport (in which case you'll need to sign in) or not (in which case you'll have to create one).

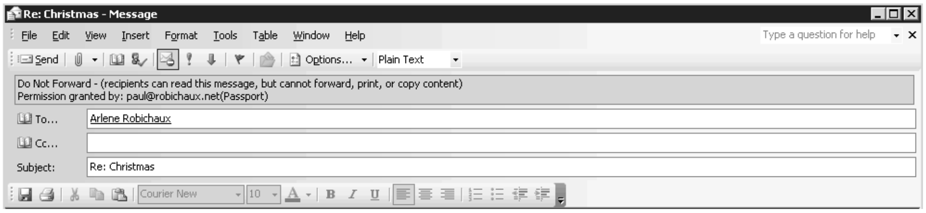
After signing in, the next question you'll be asked is whether you want a standard or temporary certificate. The Rights Management Account Certificate (RAC) is basically a PKI-based certificate issued by the Microsoft CAs that can only be used for RMS functions. If you just want to test IRM, the temporary certificate will do fine; when these certificates expire, just go through this same wizard again to get a new one. If you want a longer lived certificate, choose the Standard option instead.

After you've completed the wizard, your RAC will be downloaded and installed locally, although you won't see it in the Certificate snap-in. At that point, you're ready to start using the IRM mechanism of Outlook.

Using IRM to Protect Messages

Outlook's interface for message protection is very simple. By default, you can protect individual messages so that they cannot be forwarded, copied, or printed. Note that you either get protection against all three of these or none at all; there's no built-in mechanism to disallow, say, just printing. (As befits their document-centric nature, Word, Excel, and PowerPoint provide more granular permissions for document editing.) When you open a protected message, a recipient can see the list of his or her rights listed in the gray InfoBar above the header.

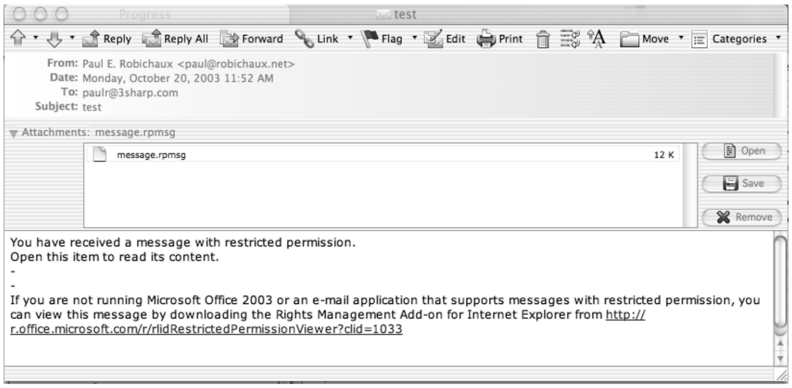
When you create a message (either by composing a new one or replying to or forwarding an existing message), you can use the File | Permissions command or the toolbar icon (which is the standard little yellow message envelope with the universal "do not enter" icon superimposed on it) to apply the Do Not Forward restriction. When you do so, the message InfoBar changes to show what restrictions are in place, as shown in Figure 13-12. Once the message is sent, recipients who open it in Outlook or Outlook Web Access will see that it's protected, and recipients using other clients won't be able to read the message contents. Instead, they'll see a text blurb telling them to use an RMS-aware client, along with a message attachment that uses the .rpmsg extension (see Figure 13-13).



Do Not Forward - (recipients can read this message, but cannot forward, print, or copy content)
Permission granted by: paul@robichaux.net(Passport)

To: Arlene Robichaux
Subject: Re: Christmas

Figure 13-12. *Protected messages are flagged when you create them.*



From: Paul E. Robichaux <paul@robichaux.net>
Date: Monday, October 20, 2003 11:52 AM
To: paulr@3sharp.com
Subject: test

Attachments: message.rpmsg

message.rpmsg 12 K

You have received a message with restricted permission.
Open this item to read its content.
-
-
If you are not running Microsoft Office 2003 or an e-mail application that supports messages with restricted permission, you can view this message by downloading the Rights Management Add-on for Internet Explorer from http://r.office.microsoft.com/r/rlidRestrictedPermissionViewer?clid=1033

Figure 13-13. *Recipients who aren't using Outlook 2003 or Outlook Web Access 2003 with Internet Explorer 6.0 and the Rights Management add-in won't be able to read the message.*

You can also use the File | Restrict Permission As command to change which set of credentials you use. This is handy if you have associated a Passport account with an RAC and simultaneously want to use one, or more, sets of corporate credentials. The credentials are associated with the user account, not with the Outlook profile.

Customizing IRM

One of the most tantalizing prospects for the combination of the Office System and the Windows RMS server is the ability to customize which rights are specified. There are a broad list of rights that apply to various pieces of the Office System, including controls that allow selective viewing, editing, saving, extracting (using copy and paste), exporting, printing, running macros, forwarding, replying, and seeing the rights associated with an object. These rights can be combined in interesting and useful ways. For example, Microsoft uses several templates internally to allow messages to be tagged with classifications like "Microsoft confidential" or "Full-time employees, read-only access." Adding more granularity to the RMS server is fairly straightforward; you have to create new templates that define the rights you want to provide and who can have them. This is easy, provided you follow the guidelines described in the online help for the RMS servers: you have to choose from the set of rights that RMS and Office both support, and the easiest way to specify who gets them is to use a universal security group defined in a domain that the RMS server has access to.

Setting up RMS client components on an individual workstation is interesting but not all that useful; the value of RMS really comes about when you can deploy it throughout an organization. There are several pieces involved in accomplishing this deployment, most of which are outside the scope of this book:

- **Installing the RMS client software** The Office System supports IRM, but there's still a separate client that has to be pushed to each participating desktop. The best way to accomplish this is to package the client with your Office deployment so that it's available on all desktops.
- **Creating rights policy templates using the Windows RMS Administration Web pages** These pages allow you to create custom sets of rights for various sets of users, then make those templates available from the RMS server.
- **Making the rights policy templates available to clients** By using the Office11.adm Group Policy template, you can specify the name of a central server from which policies should be copied and made available to all the Office System applications (not just Outlook).
- **Configuring what users can do with the RMS client** The Office policy template includes a section called Manage Restricted Permissions; by tweaking the policy settings here, you can disable the RMS user interface altogether (as you might need to if you're piloting RMS but don't want people to start randomly creating protected documents), disable the use of Passport credentials, enable or disable the use of Internet Explorer to open protected content, and control whether users must have their credentials verified every time they try to open a protected document.

The bigger issue, of course, is user training and education. It's critical that if you deploy RMS, you teach your users what can and cannot be protected. You should emphasize that RMS is primarily a technical means of more firmly enforcing the policies (like marking confidential data as such) that your company probably already has, not a way for the company to snoop on their work product.

Reaching into Outlook's Toolbox

Apart from the well-known tools discussed earlier, Outlook also has some additional security capabilities; depending on what you're doing with Outlook, these might or might not be useful to you.

Configuring the Outlook Junk Mail Filter

Outlook's Junk Mail Filter, based on work originally done by Microsoft Research, is a very handy piece of technology. In previous versions of Outlook, Microsoft shipped some junk mail filters that did an acceptable job with the style and volume of spam that was prevalent at the time. The onslaught of spam we face now, however, calls for tougher measures. Outlook's filters are designed to provide automatic, client-side filtering that works in conjunction with the Exchange Information Store (as described in Chapter 8, "SMTP Relaying and Spam Control") and perimeter filters. However, the Outlook filters give good results even when used against IMAP or POP accounts.

Note The Outlook Junk Mail Filter works for IMAP, POP, Hotmail, HTTP, and Exchange accounts. However, Exchange filtering only works when you use cached Exchange mode or delivery to a personal folder store (PST); the filters do not work with Exchange's online mode, and they don't work with third-party MAPI connectors like those from Bynari or Oracle. That raises an interesting issue, which I'll get to in a minute.

In addition to the built-in junk filter rules, which you cannot view or change, Outlook gives you another mechanism to control how mail is processed. There are three lists stored for each mailbox, either in the local PST file or the Exchange mailbox; when the lists are stored in the Exchange mailbox, they're available to the user whenever he or she logs in to Outlook or Outlook Web Access. The lists will probably sound pretty familiar:

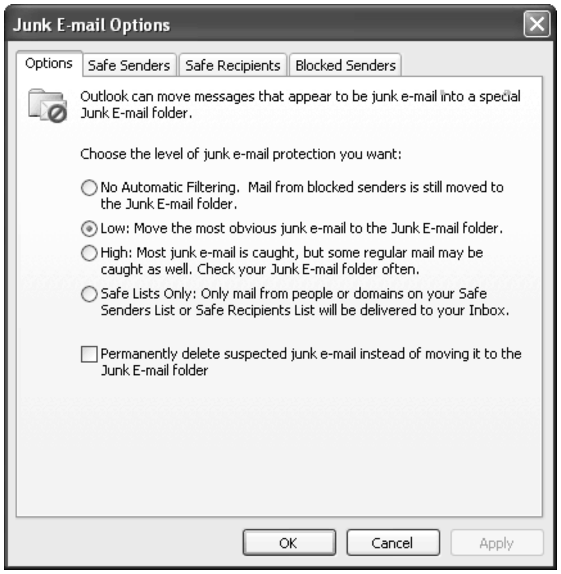
- The Safe Senders list contains e-mail addresses and domains that the user explicitly trusts. Mail sent from one of these addresses will never be flagged as junk. For example, my Safe Senders list contains microsoft.com so that mail sent from Microsoft addresses will never be flagged as junk.
- The Safe Recipients list contains e-mail addresses and domains that the user expects to receive mail from. For users who have multiple e-mail services in a single profile, adding the Exchange mailbox's address to the Safe Recipients list prevents any mail sent to that address from being flagged as junk.
- The Blocked Senders list is for people and organizations that you don't want to receive mail from. As with the Safe Recipients list, you can add individual domains or addresses to this list. Messages whose sender address or sender domain appear on this list are flagged as junk.

If you add the same sender or domain to the Safe Senders and Blocked Senders list (either accidentally or on purpose), Outlook errs on the side of conservatism and treats the message as safe.

Working with the Junk Mail Filters

Using the Junk Mail Filter is easy: as mail arrives, it's filtered, with varying degrees of aggressiveness, into the Junk E-Mail folder. You can inspect the contents of that folder at any time, deleting messages or marking them as you see fit.

The Junk E-Mail Options dialog box (see Figure 13-14) is accessible from the Junk E-Mail button on the General tab. You use this dialog box primarily to control the level of aggressiveness of the Junk Mail Filter. There are four levels:



Outlook can move messages that appear to be junk e-mail into a special Junk E-mail folder.

Choose the level of junk e-mail protection you want:

( ) No Automatic Filtering. Mail from blocked senders is still moved to the Junk E-mail folder.

[✓ Low: Move the most obvious junk e-mail to the Junk E-mail folder.]

( ) High: Most junk e-mail is caught, but some regular mail may be caught as well. Check your Junk E-mail folder often.

( ) Safe Lists Only: Only mail from people or domains on your Safe Senders List or Safe Recipients List will be delivered to your Inbox.

☐ Permanently delete suspected junk e-mail instead of moving it to the Junk E-mail folder

Figure 13-14. *The Junk E-Mail Options dialog box.*

- When you choose the No Automatic Filtering radio option, blocked senders' mail is still moved to the junk folder, but the Junk Mail Filter itself is not run, so junk messages still end up in your Inbox. Because this level doesn't block inbound spam at all, it's probably not the best choice for most environments.
- The Low option is the default setting; when the Junk Mail Filter is set to this level, "obvious" junk mail is filtered. I don't know exactly what "obvious" means in this context, but this setting does a decent job of catching most spam messages that escape common perimeter filters, and it has a very low false-positive rate.
- The High option turns the Junk Mail Filter up a notch; in this mode, Outlook is much more aggressive about mail that arouses its suspicions. As the option description in the dialog box notes, though, this mode might sidetrack some legitimate mail, too, so you should check the Junk E-Mail folder frequently.

- The Safe Lists Only option is the *ne plus ultra* of spam filters: any inbound mail whose sender isn't on your Safe Senders or Safe Recipients list will go straight to the Junk E-Mail folder. This is a great way to limit the amount of mail you have to look at, but it will probably take a while until you get the Safe lists fleshed out enough for this mode to be useful.
- The Permanently Delete Suspected Junk E-Mail check box is dangerous but useful. When you select it, Outlook removes any message that it otherwise would store in the Junk E-Mail folder. This minimizes the amount of time you have to spend cleaning up the junk folder, but it increases the risk that Outlook will delete some mail that you really wanted it to keep. In most cases, it's safer to leave this off, but if you have high spam traffic and you're comfortable with the filtering decisions of the Junk Mail Filter in Low mode, you might want to turn this option on.

As with most other Outlook 2003 settings, you can use GPOs to deploy and enforce these settings for users; look under User Configuration | Administrative Templates | Microsoft Office Outlook 2003 | Tools | Options | Preferences | Junk Mail in the Outlk11.adm GPO template.

Tweaking the Safe and Blocked Lists

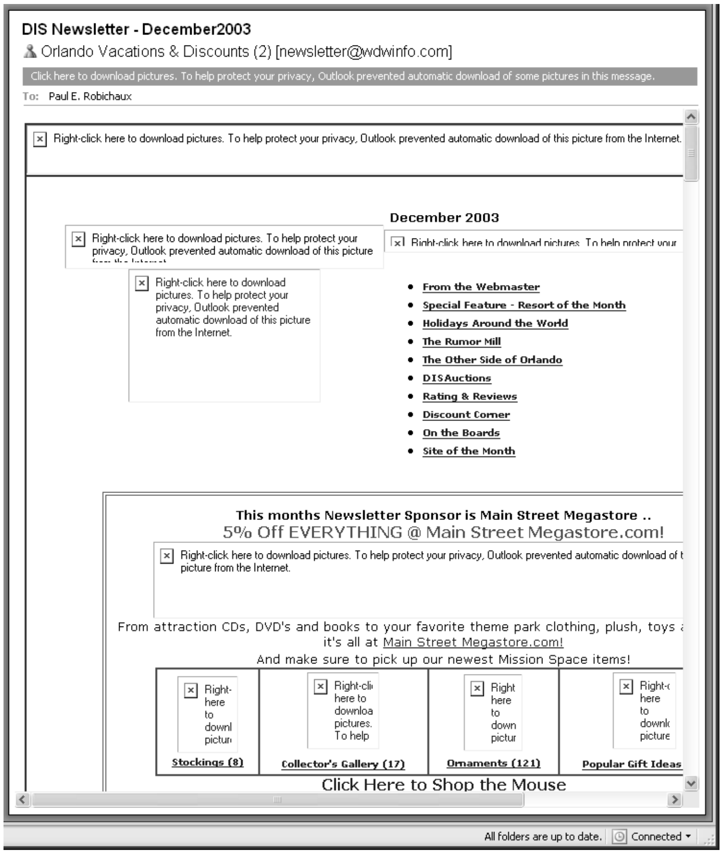
What about adjusting the filtering lists? To add a message sender or recipient address to the Safe Senders, Safe Recipients, or Blocked Senders lists, you have two choices: you can use the tabs in the Junk E-Mail Options dialog box to manage the lists, or you can right-click individual messages and use the Junk E-Mail command on the shortcut menu to add the sender or recipient address or domain to the appropriate list. The tabs give you a greater degree of functionality, because they also include buttons for importing and exporting their respective lists to disk files. This is handy because it provides a way to quickly clone one mailbox's settings to other mailboxes on a small scale (for larger scale cloning, read on).

Creating Standardized Filter Lists

You can easily create a standardized set of Safe Senders, Safe Recipients, and Blocked Senders lists and deploy them as part of your initial Outlook 2003 deployment. You'll need to create the lists on a test computer, then use the Export To File button in each list's tab to save the files with unique names. Once that's done, you can use the Office Custom Installation Wizard to package the lists for deployment with Outlook. As described in the Office Resource Kit section on deploying Outlook 2003, the Custom Installation Wizard allows you to individually specify files for these lists and whether you want Outlook's setup routine to overwrite existing lists or append the new list to whatever the user's already defined.

Controlling Automatic Image Downloads

One favorite trick of spammers is the use of *beacons* or *Web bugs*—small (usually 1×1) images embedded in HTML e-mail. When the e-mail is opened, most HTML-aware e-mail clients attempt to fetch the embedded image from a server; a savvy spammer can use the Web server's logs (combined with information embedded in the message) to track information about the user who opened the message. It's very difficult to distinguish between legitimate images embedded in mail and those that serve as beacons, so Outlook 2003 helpfully defaults to not fetching *any* images linked to remote servers in HTML mail. Figure 13-15 shows what an inbound HTML message looks like with these images turned off; users can always restore the images by right-clicking one of the placeholders and selecting the Download Pictures command.


DIS Newsletter - December2003
Orlando Vacations & Discounts (2) [newsletter@wdwinfo.com]
Click here to download pictures. To help protect your privacy, Outlook prevented automatic download of some pictures in this message.
To: Paul E. Robichaux

Right-click here to download pictures. To help protect your privacy, Outlook prevented automatic download of this picture from the Internet.

December 2003

Right-click here to download pictures. To help protect your privacy, Outlook prevented automatic download of this picture

Right-click here to download pictures. To help protect your privacy, Outlook prevented automatic download of this picture from the Internet.

- From the Webmaster
- Special Feature - Resort of the Month
- Holidays Around the World
- The Rumor Mill
- The Other Side of Orlando
- DISAuctions
- Rating & Reviews
- Discount Corner
- On the Boards
- Site of the Month

This months Newsletter Sponsor is Main Street Megastore ..
5% Off EVERYTHING @ Main Street Megastore.com!

Right-click here to download pictures. To help protect your privacy, Outlook prevented automatic download of t picture from the Internet.

From attraction CDs, DVD's and books to your favorite theme park clothing, plush, toys
it's all at Main Street Megastore.com!
And make sure to pick up our newest Mission Space items!

Stockings (8) | Collector's Gallery (17) | Ornaments (121) | Popular Gift Ideas

Click Here to Shop the Mouse

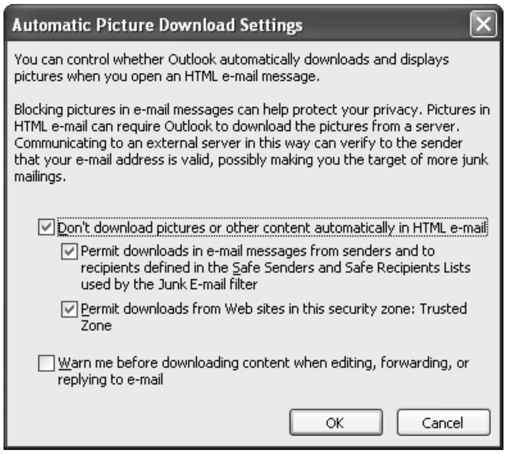
All folders are up to date. Connected

Figure 13-15. *Outlook 2003 doesn't display inline images by default.*

This behavior is controlled through the Change Automatic Download Settings button, which sharp-eyed readers might have noticed on the Security tab shown earlier in Figure 13-8. When you click this button, you'll see the Automatic Picture Download

Settings dialog box shown in Figure 13-16; you can also open this dialog box by right-clicking an image placeholder and choosing the Change Automatic Download Settings command. The options in this dialog box are pretty straightforward:

- The Don't Download Pictures Or Other Content Automatically in HTML E-Mail check box is selected by default; when it's selected, Outlook does not load images tagged with the IMG SRC tag if they point to a remote server. However, the behavior of this check box is modified by the two following check boxes.
- The Permit Downloads In E-Mail Messages check box allows you to specify that you want images embedded in e-mail from people you trust. This is a handy way to bypass the default image blocking of Outlook for your mom, spouse, mailing lists, newsletters, or other sources that you trust not to spam you with Web bugs. This option is on by default.
- The Permit Downloads From Web Sites check box controls whether Outlook downloads images where the tags point to sites in the Trusted IE security zone. Enabling this option allows Outlook to automatically download images only if the IMG tag points to a site that's in the Trusted security zone. Because you should already be careful about what sites you put in that zone, this option is on by default.
- No matter what settings you apply here, Outlook will download embedded images when you forward or reply to an HTML message that contains them. This can effectively negate the protection you got in the first place, although you probably shouldn't be responding to spammers' messages anyway. The Warn Me Before Downloading Content When Editing, Forwarding, Or Replying To E-Mail check box lets you ask Outlook to warn you before you do this. It's turned off by default.


Automatic Picture Download Settings

You can control whether Outlook automatically downloads and displays pictures when you open an HTML e-mail message.

Blocking pictures in e-mail messages can help protect your privacy. Pictures in HTML e-mail can require Outlook to download the pictures from a server. Communicating to an external server in this way can verify to the sender that your e-mail address is valid, possibly making you the target of more junk mailings.

- [✓] Don't download pictures or other content automatically in HTML e-mail
  - [✓] Permit downloads in e-mail messages from senders and to recipients defined in the Safe Senders and Safe Recipients Lists used by the Junk E-mail filter
  - [✓] Permit downloads from Web sites in this security zone: Trusted Zone
- [ ] Warn me before downloading content when editing, forwarding, or replying to e-mail

Figure 13-16. *Changing picture download settings.*

Converting Inbound HTML Mail to Plaintext

The existence of HTML mail is a sore point for many mail users, particularly those who come from a UNIX background. On one hand, HTML mail can contain pretty colors, fonts, images, and so forth. On the other hand, it takes more space to store and transfer, and scripts embedded in HTML mail can do a variety of annoying or even destructive things. Users' complaints found a sympathetic ear in the Outlook product group, so Outlook 2002 Service Pack 1 and later versions contain a feature that lets you forcibly convert all HTML mail to plaintext. Of course, this strips out all of the useful formatting, but it also renders impotent any scripts in the message, saving you from potential attacks that exploit Internet Explorer vulnerabilities. If you add a new DWORD value named `ReadAsPlain` to the *HKEY_CURRENT_USER\Software\Microsoft\Office\10.0\Outlook\Options\Mail* key, then give it a value of 1, Outlook converts HTML mail to plaintext, preserving embedded images as attachments. This doesn't affect signed or encrypted messages, but all other messages are updated as they're read. You can use this registry key in system policies or GPOs, as described in Microsoft Knowledge Base article 307594.

Encrypting RPC Traffic

RPC traffic between Outlook and Exchange Server is already compressed, and it's mostly unintelligible anyway. However, for added security (particularly for users who are using physically insecure links), you can force Outlook to encrypt RPC packets before they leave your computer. The encryption isn't as strong as the Windows VPN software, but you can use RPC encryption on your LAN or in conjunction with Microsoft ISA's MAPI RPC publishing feature—both situations where VPNs would just get in the way.

This change needs to be made to each individual client, unfortunately, although it's supported by Outlook 2000 and later versions. To force Outlook to encrypt RPCs to the server, do the following:

1. Launch Outlook.
2. Choose the Tools | E-Mail Accounts command. Verify that View Or Change Existing E-Mail Accounts is selected, and then click Next.
3. Select your Exchange e-mail account, and then click Change.
4. When the Exchange Server Settings dialog box opens, click More Settings.
5. In the Microsoft Exchange Server dialog box, click the Advanced tab.
6. Make sure that the When Using The Network check box is set, and then click OK to return to the E-Mail Accounts wizard.
7. Click Next and then click Finish.

Controlling Outlook Folder Home Pages

Outlook 2003 continues Outlook's provision of a useful but scary feature first delivered in Outlook 2000: the ability to use folder home pages so that visiting a folder automatically loads the Web page associated with that folder. This is particularly useful when used with public folders, because it allows you to associate content on an intranet (like a customer relationship management or enterprise resource planning system or other line-of-business application) with a folder. However, any scripts embedded in the page can make calls to the Outlook object model, so they can easily steal users' mail, send mail, or do a variety of other potentially undesirable things. In the normal scheme of things, this is not a huge risk. However, because anyone who can create a public folder and tie a home page to it can potentially use that ability for evil, it's a good idea to watch out.

The Outlook 2003 policy template includes a policy called Disable Folder Home Pages (under Microsoft Office Outlook 2003\Miscellaneous\Folder home pages for Outlook special folders). When you enable this policy, it automatically blocks folder home page access for all users who are subject to the policy.

Summary

Outlook has been widely criticized for security problems. In fairness, some of those could not reasonably have been predicted when the original security features of Outlook were being designed, and others result from implementation mistakes that have been fixed in service releases and hotfixes. Microsoft has worked hard to strengthen Outlook security by adding better attachment controls, complete support for S/MIME, and a range of other security features.

Additional Reading

- Outlook expert Sue Mosher maintains what must be the most comprehensive Outlook-oriented site on the Web at *http://www.slipstick.com*. In particular, her write-up of the management options for the Outlook E-Mail Security Update (see *http:// www.slipstick.com/outlook/esecup/admin.htm*) is a great summary of how to make the product do what you want it to do.
- The Microsoft Office Resource Kit site has complete information on the Outlook security package, setting up RPC over HTTP, using IRM, and other Outlook security topics. See *http://www.microsoft.com/office/ork/2003/three /default.htm*.
- For more information on the Windows RMS package, see its page at *http:// www.microsoft.com/rms*.
- To learn more than you ever wanted to know about S/MIME algorithms, protocols, and implementation considerations, visit the S/MIME Working Group of the Internet Engineering Task Force (IETF) at *http://www.imc.org /ietf-smime/index.html*.